AFRL-RY-WP-TP-2010-1159

AN AGENT-BASED MODE-CHANGE FRAMEWORK FOR FLEXIBLE WIRELESS SENSOR NETWORKS (POSTPRINT)

Paulo Martins

Chaminade University of Honolulu

JUNE 2010

AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
June 2010	Conference Paper Postprint	08 September 2006 – 31 August 2009

4. TITLE AND SUBTITLE
AN AGENT-BASED MODE-CHANGE FRAMEWORK FOR FLEXIBLE WIRELESS SENSOR NETWORKS (POSTPRINT)

5a. CONTRACT NUMBER
FA8650-05-D-1912-0007

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62204F

6. AUTHOR(S)
Paulo Martins

5d. PROJECT NUMBER
7622

5e. TASK NUMBER
11

5f. WORK UNIT NUMBER
7622110P

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Chaminade University of Honolulu
3140 Waialae Avenue
Honolulu, HI 96816

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Sensors Directorate
Wright-Patterson Air Force Base, OH 45433-7320
Air Force Materiel Command
United States Air Force

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/RYRR

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-RY-WP-TP-2010-1159

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES
Conference paper published in the Proceedings of the 24th Annual ACM Symposium on Applied Computing; conference held 9-12 March 2009 in Honolulu, HI. Paper contains color. PAO Case Number: WPAFB 08-5279; Clearance Date: 02 Sep 2008.

14. ABSTRACT
The next generation of wireless sensor networks (WSN) is required to cope with increasingly more complex applications and scenarios, involving a range of sensors supporting diverse functions such as the monitoring, detection and tracking of moving objects. These applications often exhibit changing, dynamic requirements, which may correspondingly require from the network some form of adaptive behavior. In this paper, we describe and propose a mode-change framework for wireless sensor networks. A mode-change framework allows a WSN to implement modes of operation and to change from one mode to another according to changes in the environment, thus adding more adaptability to the applications. The framework uses agents to implement one or more applications as well as to manage the transitions from one mode to another. The use of mobile agents and modes of operation confer the WSN an extra degree of flexibility, due to the ability to easily replace entire sets of agents while selectively maintaining others in operation upon demand. .

15. SUBJECT TERMS
wireless sensor networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Nivia Colon-Diaz
Unclassified	Unclassified	Unclassified	SAR	14	19b. TELEPHONE NUMBER *(Include Area Code)* N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

An Agent-Based Mode-Change Framework for Flexible Wireless Sensor Networks

Dr Paulo Martins

ABSTRACT

The next generation of wireless sensor networks (WSN) is required to cope with increasingly more complex applications and scenarios, involving a range of sensors supporting diverse functions such as the monitoring, detection and tracking of moving objects. These applications often exhibit changing, dynamic requirements, which may correspondingly require from the network some form of adaptive behavior. In this paper, we describe and propose a mode-change framework for wireless sensor networks. A mode-change framework allows a WSN to implement modes of operation and to change from one mode to another according to changes in the environment, thus adding more adaptability to the applications. The framework uses agents to implement one or more applications as well as to manage the transitions from one mode to another. The use of mobile agents and modes of operation confer the WSN an extra degree of flexibility, due to the ability to easily replace entire sets of agents while selectively maintaining others in operation upon demand.

Categories and Subject Descriptors

C.2.4 [**Computer Communication Networks**]: Distributed Systems – *Distributed applications*

General Terms

Agent, Design, Reliability, Algorithms

Keywords

Flexible Wireless Sensor Networks, Mode Changes, Mobile Agents, Coordination

1. INTRODUCTION

Wireless sensor networks have recently attracted significant attention due to their potential to bring solutions to many areas of our economy and life. As a novel technology, they combine the areas of embedded systems, sensor technology and wireless communications. This combination makes WSN a very promising technology for health, environment and military applications. A WSN consists of small sensor nodes that are low-cost, low power and multi-functional. The sensors communicate within short distances: in order for a message to be sent from a source to remote destination, intermediate nodes have to be used to relay the information, using a multi-hop transmission approach.

SAC'09, March 8-12, 2009, Honolulu, Hawaii, U.S.A.

The first key aspect of this work is the definition and the application of the notion of *modes of operation* to WSNs. Like other systems, these networks can benefit from the partitioning of an application into modes of operation. The idea of designing systems around the concept of modes of operation is fairly common and is not new: in many applications, these systems are expected to execute in several modes of operation and undergo transitions between modes in response to external events in the environment.

The second key aspect of this work is the use of *mobile agents* in lieu of static processes. In many conventional WSNs, a static special purpose process runs on each node and is able to read local sensors, execute some basic processing on the raw information and exchange messages with other peer nodes. These processes are pre-programmed (static) and deployed with the network. If reconfiguration is required, the nodes have to be physically collected and reprogrammed. This scheme is suitable for simple, static applications that do not need frequent reconfiguration, consists of a small deployment (i.e. number of nodes), over a reasonably short period of time, and performs only a small set of functions. Nonetheless, given the multi-functionality characteristic of these sensors, the use of the aforementioned method of programming does not utilize the sensors to their full capability. When reconfiguration is required, the nodes should be easily accessible and likewise easily reprogrammed. In many indoor and small-scale outdoor applications, these requirements are realistic and conventional WSNs suffice. For applications that do not fit this profile – being multi-functional, with nodes that cannot be accessed easily once deployed but do require reconfiguration – a flexible design is required.

Unlike these static processes, agents are small programs that are deployed (injected) to the network from a base station, and then can move from node to node carrying out specific tasks, such as searching for information. Due to their high mobility, the application of agents in WSN makes it a particularly attractive platform to applications that require event detection and tracking. In a generic WSN network populated with a set of agents, a mode-change model and protocol is required, one that allows the orderly and dynamic reconfiguration of the layout of agents at any time through the proper elimination of unwanted agents, the maintenance of the ones that add purpose to the application, and the redeployment of new agents which are needed but not currently present in the network.

Modes of operation are particularly suitable to WSN applications that demand flexibility, i.e. a change of functionality or behavior in response to external events. Adaptive, reactive mode-changes where the system autonomously or semi-autonomously changes mode may be a critical requirement for the next generation of

flexible WSNs, instead of an optional feature. By means of modes, it is possible to assign an application several performance levels. Each performance level can be a sub-mode of operation. For example, a trust management mode can have distinct performance layers (sub-modes): *(i)* a "strong mode", consisting of a larger number of agents, and with agents being heavier in order to deliver the maximum processing power that the application may require at a particular time, but at the expenses of more power consumption; *(ii)* a "light mode", where there are less agents deployed and agents are lighter too, with the aim of minimizing power consumption and ensuring the continuity of service, albeit at a lower quality. In between these extremes there can be any combination of performance layers offering intermediate services.

Clearly, the major disadvantage of using modes is the cost implied in changing modes, in terms of delays and power consumption. However, this can be remediated by carefully defining a mode change protocol that takes into consideration these concerns. It is also possible to minimize the number of required transitions by carefully partitioning the design into a sufficiently small number of modes. The size of a mode, or its granularity, is defined at design time and it consists in the number of agents (or tasks) that comprise that mode.

Although there has been some discussion in the literature on introducing middleware to a WSN, making it a platform for deploying multiple applications [1,3,5], there is still a lack of a framework for implementing modes and transitions in wireless sensor networks using mobile agents, such as we envisage in this work.

In this paper we propose a design framework that is as flexible as possible in order to meet the demands of next-generation WSN applications. We argue that the maximum flexibility of a WSN can be better achieved by the use of mobile agents in lieu of static processes, a modal (i.e. multi-mode) framework at the architecture level, and a parametric design of agents at the programming level.

The contribution of this paper lies in establishing a mode-change framework (concept, model and protocols) for wireless sensor networks based on mobile agents. To our knowledge, this represents an entirely novel approach to adding flexibility and more adaptation for WSNs. It also and represents a significant leap in advancing the state-of-the art for this class of systems. The study of overall design, implementation and performance analysis for this framework will be addressed in a separate publication.

The rest of this paper is organized as follows: Section 2 elaborates on the problem formulation while Section 3 addresses previous work. In section 4 we introduce the mode-change framework, and Section 5 presents a couple of motivating examples. In Section 6 we compare the proposed model with current alternative approaches. Finally, in Section 7, we summarize and offer our concluding remarks.

2. PROBLEM FORMULATION

A flexible wireless sensor network consists of a large number of autonomous agents scattered throughout the network, possibly carrying out different tasks such as event detection, tracking and monitoring.

Specific operations can be applied to agents, such as cloning and moving. An agent is a virtual machine. It may carry its state machine as it moves from node to node across the network. Agents are able to communicate in many different ways but decoupling of space and time in the communication process is a desirable feature. The most common form of decoupled communication is using associative matching based on the Linda model [4].

The reconfiguration of the type and number of agents that populate the network is crucial. Since the network responds to external events, and since conditions in the surrounding environment are dynamically changing, new agents are needed to cope with the new scenarios, and some agents become obsolete and need to be promptly removed from the network.

Assume a sensor network application evolves through four operational modes M_1, M_2, M_3 and M_4. A mode is specified as a set of agents, as shown below from sets 1 through 4. For example, mode 1 consists of agents A_1, A_2, and A_3. In order to transition from mode 1 to mode 2, agent A_1 has to be aborted (A_{1_aa}), A_{10} and A_{11} have to be fetched from the base station (A_{11_wn}), agent A_2 has to complete (A_{2_co}), and agent A_3 will run unchanged across the transition (A_{3_ua}). All the necessary mode-changing actions that need to take place to execute a cycle of transitions from mode 1 through 4 and back to 1 are represented from sets 5 through 8.

$$M_1 = \{A_1, A_2, A_3\}; \quad (1)$$

$$M_2 = \{A_{10}, A_{11}, A_3\}; \quad (2)$$

$$M_3 = \{A_6, A_7, A_8\} \quad (3)$$

$$M_4 = \{A_{3'}, A_4, A_5\} \quad (4)$$

$$MC_{1\rightarrow 2} = \{A_{1\_aa}, A_{2\_co}, A_{3\_ua}, A_{10\_wn}, A_{11\_wn}\} \quad (5)$$

$$MC_{2\rightarrow 3} = \{A_{10\_aa}, A_{3\_aa}, A_{11\_co}, A_{6\_wn}, A_{7\_wn}, A_{8\_wn}\} \quad (6)$$

$$MC_{3\rightarrow 4} = \{A_{6\_aa}, A_{7\_co}, A_{8\_co}, A_{3'\_wn}, A_{4\_wn}, A_{5\_wn}\} \quad (7)$$

$$MC_{4\rightarrow 1} = \{A_{4\_aa}, A_{5\_co}, A_{3\_ca}, A_{1\_wn}, A_{2\_wn}\} \quad (8)$$

The concern of this paper is how to dynamically deploy agents according to a pre-defined allocation of agents, such as the one given by sets 1 through 4, and how to dynamically perform the necessary transitions that allow the reconfiguration of agents, such as the ones specified by sets 5 through 8. The dynamic deployment is ensured by a mode arbiter entity, which arbitrates requests for mode changes and automatically decides (based on rules of value functions) the next mode to move to. The pre-allocation of agents is accomplished by the applicant, designer or end-user, which configures offline all the modes of operation by assigning agents in a library to modes of operation.

3. PREVIOUS WORK

There have been a few papers using the notion of modes applied to wireless sensor networks. However, in most cases modes are applied to one or more of the network layers (such as the physical, MAC or routing layers) in order to implement energy-aware protocols. For example, Cress *et al.* [2] consider three operational modes: *Sleep, Receive*, and *Transmit*. The *Receive* mode refers to the case where the transceiver is on but not transmitting, regardless of whether it is actually receiving or not.

Since we are exclusively concerned about modes at the application level, we will refrain from further discussing it.

Prehofer *et al*. [8] introduces the concept of network modes to serve different applications in an optimal way. There are three modes specified, which are the focus of the work: *Energy mode*, *Fast mode*, and *Reliable mode*. Unlike the previous example, here the implementation is accomplished through a separate 'mode control layer'. The sensor nodes used in the implementation were standard MicaZ motes from Crossbow Technology Inc. Three applications (body monitoring, health alert and fire alarm) were implemented in TinyOS. A mode is defined by *'a set of network parameters and their resulting network behavior'*. Thus, a network mode is a defined network behavior achieved by setting up specific network parameters accordingly.

Fok *et al.* [3] introduced Agilla, a middleware for WSNs that allows the programming and injection of mobile-agents. Instead of requiring data to be sent over unreliable wireless links to the location containing the computation, agents can save energy by bringing the computation to the data. Since multiple agents run in a node, multiple applications can co-exist.

In this paper, we build on previous work by (Martins) Pedro and Burns [6], where the authors developed a model for mode changes in single processor, fixed-priority preemptively scheduled systems. The model was subsequently analyzed for static, worst-case response-time guarantees. Clearly, any mode-change model for a single processor system needs to be adapted to the context of WSN's. One of the main factors is the consideration of agents in lieu of tasks and the mobility of agents in a distributed environment. Although the mode-change framework we propose can also be applied to support network layers, we are primarily concerned with application modes. A classic example of application modes in aircraft operation include but are not limited to *Take-off*, *Level flight* and *Landing.*

4. THE MODE-CHANGE FRAMEWORK

The mode-change framework is composed of the following elements:

> **Conceptual approach**: before any design and implementation work commences, it is clearly necessary to elicit all the requirements for a mode-change model; it is also crucial that the notion of modes be established in advance, in a way that it clearly and unequivocally expresses its semantics in the context of a wireless sensor network based on mobile agents;
>
> **Mode-change model:** This model defines the role and the exact behavior of agents when moving across a mode change.
>
> **Mode-change protocol:** The protocol defines the main actors (such as network nodes, base station and agents) their interactions, and the messages exchanged in order to accomplish a mode change; the mode-change protocol also defines the rules that allow the coordination of agents running across a mode change.

In the following sections we tackle each one of the framework elements in turn.

4.1 Conceptual Approach

In this paper a mode of a wireless sensor network node is defined as:

"The behavior of the network, described by a set of allowable functions and their performance attributes, and hence by a single schedule, containing a set of agents".

It follows from the definition above that a *mode change of a flexible wireless* sensor network can be defined as:

"A change in behavior exhibited by the network, accompanied by a change in the scheduling activity of the network".

The schedule of agents is the set of agents and their timing attributes such as period (T), execution time (C), and priority (P) (if any). This means that a mode change can occur by adding or eliminating one or more agents, but also by maintaining the set of agents and simply changing one of their timing attributes (such as the period, which ultimately will change the behavior of the application).

A mode-change in the WSN begins with the issuance of a mode-change request from a source (i.e. a sink node or a regular node) and ends when all nodes have completed their mode change.

The requirements of a mode-change framework for wireless sensor networks are the following:

> **Short Latency**: The transition period is a time where the WSN may be temporarily dysfunctional, or partially dysfunctional, while agents are being reallocated. Therefore it should be kept as short as possible. One example of minimization of the mode-change time follows a static approach: once a latency function is found, a genetic algorithm can be used to minimize the latency time.
>
> **Power consumption**: The need to minimize power consumption incurred in a mode-change is critical, and deserves more emphasis in systems that frequently change mode of operation.
>
> **Flexibility:** a mode-change model should allow a wide range of mode-changing scenarios, along with the specification of an arbitrary combination of agents for a given network node, no matter what the current mode is. We further discuss flexibility in Section 6.

This paper mainly tackles the requirement of flexibility. The other two requirements are deferred to a subsequent paper discussing the design, implementation and performance analysis of the proposed flexible WSN.

4.2 Mode-Change Protocol

There are a variety of types of mode changes that can be adapted to meet the specific requirements of flexible applications. This paper limits the discussion to the implementation of arbiter-based mode changes. In these types of changes, a request for a mode-change has to be arbitrated by a central entity before the system can undergo a change. The sequence of steps that characterize a change are: 1) An event is detected, requiring special attention; 2) A *mode-change request* (MCR) is generated; 3) The request is propagated to the arbiter; 4) The arbiter accepts, rejects, or defer the request; 5) The arbiter generates a reply message, the *mode-change command* (MCC), conveying the decision to the system;

6) The system begins the transition in case the mode change request is accepted.

Part of the mode-change protocol consists in disseminating the mode-change command across the network. This can be accomplished: 1) statically, where the mode-change agent follows a pre-planned path to reach affected nodes; 2) dynamically, where an algorithm specifies the nodes to be visited, or 3) following a procedure that combines both approaches.

4.3 The Mode-Change Model

The agent model consists of application agents and system agents. Application agents implement the behavior of an application (i.e. the intended functionality and the performance attributes of the services delivered) while executing in a regular mode of operation. System agents are used for the coordination or the many interactions that take place among agents during the mode change. These agents may be either inexistent or in a sleeping state between mode changes.

4.3.1 Application Agents

There are five types of application agents: 1) completed, 2) aborted, 3) wholly new, 4) unchanged, and 5) changed. These agents are classified according to how they change the behavior (i.e. functionality and performance attributes) of the application before and after the mode change. They are defined in the following paragraphs and illustrated in Figure 1.

Completed agents (A_{co}) are under execution when the mode-change command arrives and are allowed to complete during the mode change. An agent that is completed is allowed to run a few more times during the transition before it dies, typically a number between one and three executions (clearly, assuming the agent is a periodic one). This allows for final extra measurements before the application changes mode. From the application perspective, the completed agent set is useful to model the functionality that has safety-critical implications to the system. The system executes the mode change while also delivering the old functionality in order to leave the system in a safe condition. Also, agents that have started may need to finish for data consistency and future executions.

Aborted agents (A_{aa}) are abandoned at the time a mode change command arrives at a node. An aborted agent terminates its current execution and does not run anymore. From the application point of view, the functionality provided by this class of agents is no longer necessary at the start of a mode change and thus can be discarded. In terms of implementation, this class of agents can be seen as a special case of an old-mode completed when the number of extra executions during the transition is zero. For the sake of the system's performance, they need to be quickly eliminated in order to release computational resources for other agents.

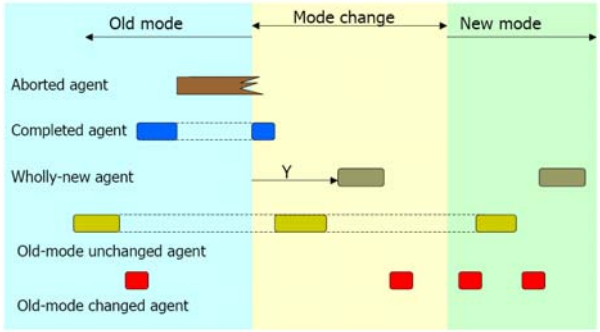

Figure 1. The mode-change model for WSNs. time→

Wholly new agents (A_{wn}) implement the new behavior required by the system. Unlike the other new mode agents, they do not change or replace any old-mode agent. These agents are injected to the network from a base station and move to a target node where they need to execute. In case there is more than one target node, cloning this agent after injection is often required.

Unchanged agents (A_{ua}) execute before, during and after the transition is complete. This class of agents preserves all their timing attributes and execution code throughout the mode change. The behavior that these agents implement is required in the old mode and extends beyond the transition.

Changed agents (A_{ca}) implement the system's changed behavior. Changing one or more of its parameters changes the behavior of the agent. The period (if periodic agents), execution time, priority (if a priority-based system), execution code, or a combination of these factors may be altered in such a way that the new-mode agent differs from the old-mode version. Note that the difference is only marginal (e.g. parametric), to the extent that it does not qualify them to be modeled as wholly new agents. Changed agents add another level of flexibility to the WSN because they are parameterized. A small set of parameters may be adjusted so that the agent performs a repertoire of correlated functions. It also consists a suitable time and power-saving strategy when the cost of bringing a wholly new agent from the base station exceeds the network requirements.

Completed agents and aborted agents are generally referred to as *"old-mode agents"*, since they begin execution in the old mode. Wholly new agents, changed agents and unchanged agents are referred to as *"new-mode agents"*, as they begin execution during the transition and through the new mode. Both changed and unchanged new-mode agents are preceded by a corresponding old-mode version.

All new-mode agents may begin execution during the transition with a delay Y after the mode-change command arrives in the system. This offset may be necessary in order to avoid or at least minimize the critical instant, e.g. a time interval where the system may experiment a transient overload. The overload is due to the mutual interference of old-mode agents scheduled to finish their execution during the transition and the new-mode agents, which attempt to run for the first time.

The model above provides a useful abstraction that allows the coordination of multiple agents across a mode change. Clearly, these agent roles are temporary and exclusive to a particular mode

change: an agent may execute across several mode-changes during its lifespan, and its role varies from transition to transition. For example, in Figure 1, the unchanged agent in the current transition may become a completed agent in the subsequent one.

4.3.2 System agents

System agents support exclusively the coordination of agents across the mode change; they do not run any application code. There are two types of system agents as defined below:

- **The mode-manager agent (A_{MM}).** This agent is responsible for the dissemination and implementation of a *mode-change command* (MCC) across the network. The mode manager carries with it specific instructions, using a coordination language (e.g. Linda), on how to configure a node in preparation for the new mode of operation. In many cases, including applications where the network is homogeneous (nodes are identical), one agent may be enough to accomplish a mode change. In more complex scenarios, it may be required a number of mode-manager agents to orchestrate the transition and achieve the desired coordination among agents. This agent is injected into the network before the mode-change. It moves or clones itself to the nodes requiring a mode change. These agents die once they have completed their task, i.e. when the transition is progressing towards completion, thus releasing resources back to the system. It should be noted that the indication of Linda as a possible coordination language used is because these agents require some time decoupling in their operation. Upon arrival in a node, the mode-change command (MCC) is issued but the local agents may not respond to it immediately (depending on the type of agent). However, it is unreasonable for the mode-manager agent to be forced to wait to a local response before continuing and cloning itself. Hence it can issue the command and continue its main task of communicating the change to other nodes.
- **The relay agent (A_{RE}).** This agent relays messages from application agents to the base station. The path from the source of information (sender) to the sink node may need one relay agent per node. It works by simply receiving a message from a neighbor agent and forwarding it to the next hop along the path to the destination node. This agent is usually not necessary when a "producer" agent moves to the sink node carrying the information needed by the "consumer" (i.e. application). It may not be needed when the network is equipped with a suitable routing layer and messages can be sent anywhere in the network to the sink/base station. Without this provision, a protocol where an application agent just reads the information but does not move away from its node may need to use relay agents to send the information to the application.

One of the most critical performance attributes (along with power consumption) in a flexible wireless sensor network is the *latency* of a mode change. The latency depends at least on network attributes such as topology, number of agents and network exploration algorithm. A complete definition of latency in a node follows:

"The latency of a mode change in a node is the time interval that begins with the arrival of a mode-change command (MCC) at a node and ends when all old-mode agents have terminated and all new-mode agents have run at least once."

From the discussion in Section 4, it follows that:

"*The latency of a mode change in a WSN is the time interval that begins with the issuance of a mode-change request (MCR) from an arbitrary node and ends when all nodes have completed their own transitions".*

Therefore, latency of a mode-change in the WSN consists in the accumulative delays incurred by the mode-change request, its arbitration delays by a mode-manager entity, the mode-change command travel time, and the longest mode-change delay among all changing nodes. Given that WSN normally form a scale-free network, the mean distance between any two nodes tend to be small and hence the time it takes for a message to reach the sink tend to remain short.

5. EXAMPLES

One motivating example is illustrated by an unmanned aerial vehicle (UAV) application, where sensor nodes are deployed in a broad area along the body of the aircraft. The WSN system design is structured around at least four major modes of operation: 1) **Vehicle health inspection mode:** agents deployed in this mode perform tasks such as detection and tracking of leakages, overheating and damage assessment; 2) **Navigation control mode:** In this mode, navigation agents are deployed in order to communicate with base stations and acquire information about the status of the path. For example, a UAV can receive advance notice (alarm) from a node on the ground indicating the existence of a hazardous area ahead, which prompts the redirection of the UAV to a safer path or location; **3**) **Distributed radar mode:** Specific agents can be tasked and deployed to collaborate and acquire information for the distributed radar application. **4) Automatic Target Recognition mode (ATR):** In this mode agents are tasked to read the onboard cameras and convey image frames from a source to a sink node.

An unexpected circumstance may arise years after the initial operation of the aircraft, such as the specific need to monitor mechanical stress in certain areas, which was not foreseen or planned in advance. Whereas it is necessary to acquire stress status during certain phases of operation, the regular health inspection, navigation control and distributed radar functions cannot be completely abandoned. Thus these functionality sets are multiplexed in time by configuring them as operational modes. As long as the sensor capability is provided, the end-user may configure a mode by selecting a set of agents. These new-mode changed agents can be quickly formed by assigning them a value that specifies the stress sensor as the new standard input. Alternatively, it is also possible to choose from a library a preconfigured stress-management mode, and further deploy it on the fly.

Note that even in the case where the designers could have anticipated the need for a stress management function, it would likely not fit in the limited memory of WSN nodes along with the other functions in a conventional, (i.e. non-flexible) WSN. The inclusion of a new *'stress management mode'* and the new transitions from and to it is something that can be easily achieved

with a GUI for deploying applications designed around modes. In addition to that, as sensor nodes are to be embedded in the fuselage/structure of the aircraft, the physical access to nodes to allow reprogramming is not a viable option. These three characteristics discussed above, i.e. unforeseen scenarios, the limited resource constraints of nodes, and inaccessibility of a node once deployed, are clearly very pervasive for many WSN real-life applications, rendering conventional WSNs as impractical, while making a flexible framework such as the one presented here an interesting proposition (if not a mandatory one).

Clearly, there are many possibilities in terms of how transitions are initiated and managed in this application: the transition from mode to mode can be triggered by a regular node, by a cluster node or by the sink node. It can also be a time-triggered or event-triggered transition, regular or irregular, depending of the application scenario.

UAV modes may be arranged hierarchically: In the example above, each mode may be broken down into two sub-modes: detection and tracking. In detection sub-mode, detection agents are deployed looking for an abnormal event. Once an event is detected, the network moves to a tracking sub-mode, where tracking agents are deployed and detection agents are eliminated. Tracking agents swarm around the area affected by the abnormal event, giving a precise indication of its location. This arrangement of sub-modes not only adds more structure to the application but it also works as a suitable power-saving strategy, due to the fact that, as pointed out by Fok *et al.* [3], detection agents are lighter than tracking agents.

6. DISCUSSION

Table 1 compares the proposed model with three alternative approaches, according to their level of flexibility. Five attributes are addressed: programmability, modularity, scalability, complexity and adaptability. These attributes, when combined, provide an indication of the overall degree of flexibility achieved by a WSN.

Programmability refers to how easy it is to reprogram the application should the current application no longer satisfy the changing requirements.

Modularity refers to the structuring of the application (i.e. architecture) around behavioral blocks that implement a certain functionality and/or performance feature of the network.

As the name implies, ***Scalability*** of the application is related to how large the application can grow if the network has to cope with a wide range of dynamic scenarios over an extended period of time.

Complexity consists in how easy is to deploy an application that deals not only with one or two types of sensors, but a wide array of sensors. They will be used to perform not one or two monitoring tasks, but a variety, usually associated with tracking of a moving target. These sensors may all be used at one time, or individually at some other times in order to minimize power consumption. These require a large number of processing units (tasks, processes or agents) of distinct types.

Adaptability (also automaticity) refers to the capability of the WSN to automatically self-configure, i.e. to recognize an event, relate it to a mode of operation, and deploy a complex configuration of agents (mode). As application requirements change, some agents need to be completed or aborted; some have to have their execution extended over a period of time, and some need to be changed while others have to be re-injected. One key element in achieving adaptation is the *mode-change arbiter* in centralized mode-change protocols. For systems that do not rely on a central entity, adaptation can be implemented by means of *self-organization*.

We also identify four classes of WSN systems according to their level of flexibility:

Conventional WSNs: These are *'all-in-one-mode'* systems. In these systems, static processes (i.e. processes or tasks without migrating and cloning capabilities over the distributed WSN infrastructure) interact exclusively by means of message exchanges. Most WSNs reported in the literature fall under this class.

WSNs based on mobile agents: Like conventional systems, these systems do not implement modes of operation, but are designed around the concept of mobile agents. A typical example is the Agilla middleware developed by *Fok et al.* [3].

Strictly Modal WSNs: These types of systems are designed around the concept of modes, without the support of mobile agents. Usually the number of modes is small and all the modes of operation are well known before deployment [8].

Flexible WSNs: These networks, as proposed in this paper, go one step further in terms of flexibility by combining mobile agents and operational modes into a single framework.

As shown in Table 1, *conventional WSNs* are hard to reprogram and offer little or no modularity at the application level. They can be used for simple applications that are not intended to grow with time. These systems are not designed to meet the requirements of applications that demand a greater extent of adaptability. However, through the proper parameterization of the processes than run on these networks, it is possible to address complexity and adaptation on a limited basis (i.e. by changing the value of function parameters).

Agent based WSNs, offer the advantage of *reprogrammability* over conventional systems. They represent an intermediate solution between conventional systems and truly flexible WSNs. As new agents can be designed to meet new applications, these networks allow an application to scale well. However, they only marginally can handle more complex applications requiring dynamic adaptation. One would have to manually select the new agents to be injected, or manually terminate (i.e. abort) agents, but with little or no provision to replace an old-mode agent by an arbitrary new-mode one, or to request for a new agent from the field upon a changing application scenario.

Strictly modal systems are hard to reprogram and difficult to scale once deployed. They cannot handle complexity well because of the well-known resource limitations of WSNs. Their use of modes constitutes a modular design per se. As these systems can change modes in response to external events, they do offer some level of adaptability. However, these models suffer from poor scalability. They are well suited for simple applications with a few modes

where the application does not evolve over time. This is particularly true for small deployments of WSN.

In contrast with these systems, *Flexible WSNs* offer all the benefits of programmability and scalability inherited by the explicit use agents, as well as the benefits of modularity and adaptability conferred by the use of modes.

They can be used in a variety of applications, from simple to large and complex deployments; they are particularly useful in settings in which there is not much knowledge in advance of what the different modes of operation will be. Once the WSN is deployed and more knowledge is acquired, new application scenarios may unfold requiring new modes, which could not be anticipated at design time due to the "evolutionary" character of the application over a long-term deployment.

Flexible WSNs are *modular* in design. Modes provide a suitable approach to organize a design in layers. Therefore, each mode can be seen as one module in the control and functional hierarchy. The modularity achieved with modes ensures benefits such as easier engineering, deployment and maintenance. This modularity also facilities the *separation of concerns*: An *all-in-one mode* system is designed with all scenarios embedded in one module. Modes allow the network to specialize and optimize its resources for a particular phase of operation, scenario or use-case, by choosing the configuration of agents that best suits the application for that particular time interval.

The model proposed offers the advantage of *scalability* of the application. Virtually, an infinite number of modes can be accommodated, because they are downloaded upon demand, and they "overwrite" the previous mode. The sink node is used to "extend" the memory available in a WSN. Modes of operation can be compared to features such as paging in operating systems (OS). The memory available at the sink node is the equivalent of virtual memory in OS's. The application running in a CPU (i.e. WSN) requires a new page (i.e. a new mode of operation) to be loaded from secondary memory (i.e. laptop memory) to primary memory (WSN's distributed memory), for example, once the current one has completed.

The basic idea behind modes is that the combined size of the application when all modes are combined as a unit may exceed the amount of physical memory available for it in a WSN. This is particularly relevant in WSN's, where available memory is severely restricted. For example, a mode-manager entity keeps those parts of the application currently in use in the WSN, and the rest on a sink node. A 2GB application can run on a WSN by carefully choosing which 4kb to keep in a WSN node at each phase of operation (mode), with pieces of the application (modes) being swapped between the sink node and the WSN as needed.

This example shows that, in general, while modes are usually associated with an operational phase, this does not need to be always the case: modes can be used merely as a way of partitioning a large application into smaller applications that fit into the memory of a WSN node. Note that the comparison between modes and pages is not intended to be a perfect one: both differ, for example, in the granularity (page size *versus* number of agents) and in the frequency of switching. Page sizes may be smaller than modes, and pages are switched usually much more frequently than modes. However, the similarities are close enough to illustrate the idea of modes as a strategy for extending memory.

Flexible WSNs facilitate the deployment of the next generation of *complex applications*. Once a new application scenario is identified, hence correspondingly requiring a new set of agents, these agents do not need to be selected manually and individually by the user. The end user deploys a pre-configured mode, which will ultimately cause the injection of all agents implied by that mode. Modes and their transitions can also be labeled as regular, irregular, time or event-triggered. With the support of a modal engine (i.e. mode arbiter), it is possible then to process these types of modes and mode changes automatically, without user intervention, or at least with minimal user intervention.

Flexible WSNs are also *adaptable* in the sense that they can detect events in the surrounding environment and react by automatically (or semi-automatically) deploying a new mode (as mentioned before, with the help of a modal engine software running on a sink node). They are also adaptable to applications that exhibit evolutionary stages over time. This aspect was discussed in the previous section (section 5) through the example given.

Table 1. Type of WSN Design

Feature	Conventional	Agent Based	Strictly Modal	Flexible
Programmability	×	✓	×	✓
Modularity	×	×	✓	✓
Scalability	×	✓	×	✓
Complexity	⊙	⊙	×	✓
Adaptability	⊙	⊙	✓	✓

⊙ = limited × = not met ✓ = met

The model proposed is comprehensive and general. It is expected that some applications will require only a subset of the model, for example, with no need for completed agents, or no need for changed agents. However, it was our goal from inception to provide a model that is generic enough to embrace most mode-change scenarios, and that can be easily instantiated (i.e. simplified) for the less demanding applications.

7. SUMMARY AND CONCLUSIONS

In this paper we introduced a framework that allows the integration of modes of operation in wireless sensor networks (WSNs). The framework consists of a model and a mode-change protocol that enables the easy deployment of applications that require flexibility, and a repertoire of different behaviors (i.e. functionality and its performance attributes). It also includes a working definition of the notion of modes and their transitions in WSNs based on mobile agents.

This framework combines the idea of operational modes and mobile agents to further increase the adaptability of a wireless sensor network: a modal, flexible network can adapt to the external environment by changing from one mode of operation to another. Transitions from mode to mode often imply in adding (or deploying) wholly new agents from a sink node, aborting or

completing the execution or other agents, or modifying the execution of agents according to a mode-change model and protocol.

As a future work, we aim at working with the design and implementation of the framework, which may then be used to experiment flexible wireless sensor networks with real world applications.

In conclusion, we have demonstrated that the mode-change framework introduced in this paper lays a solid ground to the design and implementation of flexible wireless sensor network; it also provides a conceptual background to the study of mode changes in more complex applications such as those in real-time and embedded systems.

8. ACKNOWLEDGMENTS

This work was supported by the Wright-Patterson Air Force Research Laboratory, sub-contract CHAM 06-S567-07-C2.

9. REFERENCES

[1] Boulis, A., Han, C.-C., And Srivastava, M .' Design and implementation of a framework for efficient and programmable sensor networks. In Proc. of MobiSys. USENIX, 187-200.

[2] Cress, C., Yang, S. and Sudit, M., "Energy Efficient Data Retrieval for Networked Multi-mode Wireless Sensors", PE-WASUN 2004.

[3] Fok, C.-L., Roman, G.-C., and Lu, C.. Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications". *International Conference on Distributed Computing Systems (ICDCS'05), June 2005.*

[4] Gelernter, D, "Generative Communication in Linda", *ACM Transactions on Programming Languages and Systems* 7, 1 (January), 80-112.

[5] Levis, P. And Culler, D. 'Mate: a tiny virtual machine for sensor networks', In ASPLOS-X: Proceedings of the 10th international conference on architectural support for programming languages and operating systems. ACM Press, New York, NY, USA, 85-95.

[6] Martins Pedro, P. Burns, A. 'Schedulability Analysis of Mode Changes in Flexible Real-Time Systems'. In Proceedings of the 10th Euromicro Conference on Real Time Systems, Berlin, Germany, 17-19th Jun/1998.

[7] Martins Pedro, P. 'Implementation of Modal Wireless Sensor Network'. Internal Report CUH-CS-R08-11, Department of Computer Science. Honolulu, Hawai'i, USA.

[8] Prehofer, C Hurler B, Wei Q and Zittebart, M., "A Framework for Network Mode Control in Wireless Sensor Networks" Telematics Technical Reports ISSN 1613-849X, University Karlsruhe, Dec/2005.